

João Henrique de Araújo Kröger

Zero-shot Learning Optimal Power Flow with Graph Neural Networks

São Paulo, SP

2023

João Henrique de Araújo Kröger

Zero-shot Learning Optimal Power Flow with Graph Neural Networks

Final monography presented to the Department of Computer Engineering and Digital Systems of the *Escola Politécnica da Universidade de São Paulo* to obtain the Engineer's Degree.

Universidade of São Paulo – USP

Escola Politécnica

Department of Computer Engineering and Digital Systems (PCS)

Supervisor: Prof. Dr. Anna Helena Reali Costa

Co-supervisors: M.Sc. Allan Santos; Eng. Marcel Rodrigues de Barros

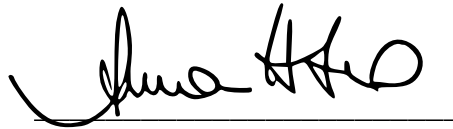
São Paulo, SP

2023

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

São Paulo, 12 de dezembro de 2023.

De acordo,



Anna Helena Reali Costa

Catálogo-na-publicação

Kröger, João Henrique
Zero-shot Learning Optimal Power Flow with Graph Neural Networks / J.
H. Kröger -- São Paulo, 2023.
47 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São
Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Optimal Power Flow 2.Graph Neural Networks 3.Zero-shot learning
I.Universidade de São Paulo. Escola Politécnica. Departamento de
Engenharia de Computação e Sistemas Digitais II.t.

Acknowledgements

First and foremost, thanks to my Lord and Saviour Jesus Christ, who has been with me through all the good and difficult times. All glory be to Him!

I am immensely grateful to my parents Renato and Márcia and to my brother Gabriel for their unconditional love and for their support in all of my decisions. I would also like to thank my aunt Lena and my grandmothers, Lila and Maria, for their unrelenting prayers and their kind words of affection.

My gratitude also extends to all of my friends, who helped bring joy and lightness to my life even in trying moments.

Lastly, I am especially thankful for the guidance of my advisor Anna Reali, as well as of my co-advisors Allan Santos and Marcel Barros, who gracefully accompanied me and helped me finish this work.

Resumo

Em sistemas de potência, o *Optimal Power Flow* (OPF) é um dos problemas de otimização mais relevantes. Seu objetivo é minimizar o custo total de geração em uma rede de energia (o qual depende dos custos unitários de cada gerador), e, ao mesmo tempo, satisfazer uma dada demanda e um conjunto de restrições operacionais. Abordagens tradicionais para a resolução do problema de OPF, como o método do ponto interior, frequentemente são computacionalmente caras e não escalam bem à medida que o tamanho da rede aumenta. Portanto, como uma maneira de superar essas limitações, este trabalho explora o uso de *Graph Neural Networks* (GNNs) como um método alternativo para resolver o problema do OPF. Ao aproveitar a estrutura de grafo inerente das redes de energia, as GNNs podem modelar, de forma efetiva, as complexas dependências e interações entre diferentes componentes, como *buses*, geradores e linhas de transmissão, e usar essas informações para determinar a alocação de potência ótima na rede. Essa representação baseada em grafos aprimora a capacidade de generalização do modelo, permitindo-o lidar com diversas topologias, tamanhos e configurações de rede, e possibilitando sua aplicação mesmo em cenários *zero-shot*, nos quais, durante a inferência, o modelo deve avaliar amostras de redes que não haviam sido observadas anteriormente. A arquitetura da GNN proposta é baseada em um mecanismo de troca de mensagens, no qual os nós trocam mensagens que contêm informações sobre seu estado e sua vizinhança. O treinamento é realizado de forma não supervisionada, com a violação das leis físicas e o custo sendo minimizados diretamente. Além disso, as restrições são aproximadas por meio de funções de penalização que punem violações. O modelo baseado em GNNs foi testado em várias redes de tamanhos e complexidades variadas, variando de menos de 10 a mais de 2000 *buses*. A avaliação experimental demonstra que ele pode ser usado como uma abordagem viável para resolver o problema de OPF, sendo capaz de gerar soluções comparáveis às dos *solvers* padrão, enquanto leva significativamente menos tempo para executar os cálculos — com uma diferença de 3 ordens de magnitude no caso mais complexo. Além de sua melhor escalabilidade, o modelo GNN também exhibe performance satisfatória mesmo em redes que não haviam sido observadas durante o treinamento (propriedade de *zero-shot*).

Palavras-chave: Graph Neural Networks. Optimal Power Flow. Aprendizado *zero-shot*.

Abstract

In power systems, optimal power flow (OPF) is one of the most relevant optimization problems. It aims to minimize the total generation cost in a power grid (which depends on the unit costs of each generator) while satisfying a given demand and a set of operational constraints. Traditional approaches for solving the OPF problem, such as the interior point method, are often computationally expensive and do not scale well as the size of the grid increases. Hence, as a way of overcoming these limitations, this thesis explores the use of Graph Neural Networks (GNNs) as an alternative method to solve the OPF problem. By leveraging the inherent graph structure of power grids, GNNs can effectively model the complex dependencies and interactions among different components such as buses, generators and transmission lines, and use this information to determine the optimal power allocation in the grid. This graph-based representation enhances the generalization capability of the model, enabling it to handle diverse grid topologies, sizes and configurations, and allowing it to be applied even in zero-shot scenarios, where, at inference time, the model has to evaluate samples from grids that had not been observed before. The proposed GNN architecture is based on a message passing mechanism, in which nodes exchange messages that contain information about their state and their neighbourhood. Training is performed in an unsupervised manner, with the violation of physical laws and the cost being minimized directly. Furthermore, the constraints are approximated by means of penalty functions that penalize violations. The GNN-based model was tested on several grids of varying sizes and complexities, ranging from under 10 to over 2000 buses. The experimental evaluation demonstrates that it can be used as a viable approach for solving the OPF problem, as it can output solutions comparable to those of standard solvers while taking significantly less time to execute the computations — with a difference of 3 orders of magnitude in the most complex case. Besides its better scalability, the GNN model also exhibits the capability of performing satisfactorily even on grids that had not previously been seen during training (zero-shot property).

Keywords: Optimal Power Flow. Graph Neural Networks. Zero-shot learning.

List of Figures

Figure 1 – Block diagram representation of Graph Neural Networks	19
Figure 2 – Gantt diagram of the proposed work packages	24
Figure 3 – Block diagram of the proposed GNN architecture	28
Figure 4 – Comparison of active power output of the GNN and of the IPS	38
Figure 5 – Comparison between proposed GNN model and IPS	38
Figure 6 – Relative generation cost of a model trained on multiple cases	39
Figure 7 – Computation time of a model trained on multiple cases	40
Figure 8 – Relative generation cost of a model trained on <i>case300</i>	40
Figure 9 – Computation time of a model trained on <i>case300</i>	41
Figure 10 – Relative generation cost of a model trained on <i>case2383wp</i>	41
Figure 11 – Computation time of a model trained on <i>case2383wp</i>	42

List of Tables

Table 1 – Model hyperparameters for each power grid	36
---	----

List of abbreviations and acronyms

DNN	Deep Neural Network
GNN	Graph Neural Network
GNS	Graph Neural Solver
GSO	Graph Shift Operator
GSP	Graph Signal Processing
IP	Interior Point
IPS	Interior Point Solver
LSTM	Long Short-Term Memory
MPNN	Message Passing Neural Network
NN	Neural Network
NR	Newton-Raphson
OPF	Optimal Power Flow
PF	Power Flow
RNN	Recurrent Neural Network
SDP	Semidefinite Programming
TSO	Transmission System Operator
WP	Work Package

List of symbols

V	Bus voltage magnitude
θ	Bus voltage phase
P_d	Bus active power demand
Q_d	Bus reactive power demand
G_s	Bus shunt conductance
B_s	Bus shunt susceptance
V^{min}	Minimum allowed bus voltage magnitude
V^{max}	Maximum allowed bus voltage magnitude
r	Power line resistance
x	Power line reactance
b	Power line charging susceptance
τ	Transformer off nominal ratio
θ_{shift}	Transformer phase shift angle
$\Delta\theta^{min}$	Minimum allowed voltage phase difference
$\Delta\theta^{max}$	Maximum allowed voltage phase difference
V_g	Generator voltage magnitude set point
P_g	Active generator power output
Q_g	Reactive generator power output
P^{min}	Minimum allowed active generator power output
P^{max}	Maximum allowed active generator power output
Q^{min}	Minimum allowed reactive generator power output
Q^{max}	Maximum allowed reactive generator power output
P_i	Active power injection at bus i

Q_i	Reactive power injection at bus i
P_{global}	Global active power consumption
Q_{global}	Global reactive power consumption
P_{joule}	Global active power consumption
ΔP_i	Local active power imbalance at bus i
ΔQ_i	Local reactive power imbalance at bus i
m_i	Latent message of bus i
ϕ	Message function of the GNN
L_v	Voltage magnitude update function
L_θ	Voltage phase update function
L_m	Latent message update function
L_p	Active generator power output update function
L_q	Reactive generator power output update function

Contents

1	INTRODUCTION	13
1.1	Motivation	13
1.2	Goals	15
1.3	Justification	15
1.4	Organization of the Manuscript	15
2	THEORETICAL FOUNDATIONS	17
2.1	Graph Neural Networks	17
2.2	Power Grid Modelling and the OPF Problem	19
3	METHODOLOGY	23
4	REQUIREMENTS SPECIFICATION	25
5	DEVELOPMENT	27
5.1	Used Technologies	27
5.2	Project and Implementation	27
5.2.1	Model Architecture	28
5.2.1.1	Initialization	28
5.2.1.2	Global Power Consumption Calculation	29
5.2.1.3	Local Power Imbalance Calculation	29
5.2.1.4	Message Passing Mechanism	30
5.2.1.5	Neural Network Update	31
5.2.2	Training	32
6	EVALUATION	35
6.1	Dataset Generation	35
6.2	Evaluation Metrics	35
6.3	Evaluation Results	36
6.3.1	Model Validation	37
6.3.2	Evaluation on Standard IEEE Power Grids	39
6.3.3	Scalability Evaluation	41
7	FINAL REMARKS	43
7.1	Conclusion	43
7.2	Contributions	43
7.3	Future Perspectives	44

BIBLIOGRAPHY 45

1 Introduction

In power systems, Power Flow (PF) analysis constitutes an integral part of the operation, planning, maintenance and control of energy transmission networks (SALAM, 2020). In simple terms, the power flow study aims at finding the steady-state operating characteristics of a power system, namely the voltage magnitude and phase at all buses of the grid, given a particular topology as well as the active and reactive powers of the loads and generators (GRAINGER; STEVENSON, 1994). By carrying out this analysis for a variety of scenarios, corresponding to e.g. different power injections, secure and reliable operation of the system can be ensured (FAN et al., 2019).

One instance of the power flow method, the so-called Optimal Power Flow (OPF) problem, attempts to not only find the steady-state values of the system variables but also determine the optimal power output of the generators within the grid in order to meet a given demand while satisfying a set of operational constraints. Optimality is dependant on the unit costs of each generator and refers to the total energy production cost for a given scenario. Due to the non-linear, sinusoidal nature of AC power flow, this optimization problem is non-convex and, in most cases, hard to solve (CAIN; O'NEILL; CASTILLO, 2012). Indeed, OPF has been proven to be NP-hard (BIENSTOCK; VERMA, 2019). Thus, considering the importance of the OPF problem to the operation of power grids, extensive research has been conducted in this field in order to find ways to efficiently solve this problem.

1.1 Motivation

Various approaches have been proposed as a way to address the general intractability of the OPF problem. Among them, one of the most common is to solve a linear proxy of the problem by means of small angle approximations (commonly referred to as DC-OPF). This method, however, fails for heavily loaded networks since differences in voltage angles become large (CHATZIVASILEIADIS, 2018). Apart from that, convex relaxation techniques, such as quadratic, second-order cone programming and Semidefinite Programming (SDP) relaxations, have been extensively studied (ZOHRIZADEH et al., 2020) and can be applied so as to deal with the non-convexity of the OPF problem and provide more general optimality guarantees. Despite that, such relaxations often lead to infeasible or sub-optimal solutions for certain grid topologies (LOW, 2014).

Recently, several machine learning-based methodologies have been put forward as an alternative to solve a wide range of problems in the power systems domain (OZCANLI; YAPRAKDAL; BAYSAL, 2020). Typically, machine learning models can evaluate the

state of a grid much faster than traditional power grid analyzers and can handle complex nonlinear relationships more flexibly. In particular, extensive research has been conducted on the use of machine learning techniques to compute the PF (DONNOT et al., 2018; DONON et al., 2020; LOPEZ-GARCIA; DOMÍNGUEZ-NAVARRO, 2022; HANSEN; ANFINSEN; BIANCHI, 2023) and the OPF (PAN; ZHAO; CHEN, 2019; SINGH; KEKATOS; GIANNAKIS, 2022; OWERKO; GAMA; RIBEIRO, 2020; OWERKO; GAMA; RIBEIRO, 2022) of electrical grids.

Given the broad scope of the machine learning field, it is unsurprising that diverse architectures have been proposed to solve tasks related to PF and OPF. For instance, numerous works focus on designing deep neural networks (DNNs) that can output solutions to these problems (PAN; ZHAO; CHEN, 2019; SINGH; KEKATOS; GIANNAKIS, 2022). Nonetheless, it should be emphasized that such models struggle with changes in the system topology and are more prone to overfitting, which limits their generalization capability (HANSEN; ANFINSEN; BIANCHI, 2023). In light of this, other studies have promoted the utilization of Graph Neural Networks (GNNs) as an alternative method of performing PF and OPF (DONON et al., 2020; HANSEN; ANFINSEN; BIANCHI, 2023; OWERKO; GAMA; RIBEIRO, 2020; OWERKO; GAMA; RIBEIRO, 2022). GNNs are neural network models that exploit the topology of the underlying graph structure to implement localized computations, making them decentralized in nature. Therefore, in principle, GNNs can be applied to grids with different sizes and topologies. This can be especially interesting for power system analyses, considering that electrical grids can naturally be modelled as a graph, where nodes correspond to buses and edges represent transmission lines.

Aside from that, approaches predicated on machine learning ideas also display diversity in how the models are trained. More specifically, supervised learning has been used to train GNN models that imitate the output of a particular power grid solver, both for PF (HANSEN; ANFINSEN; BIANCHI, 2023) and for OPF (OWERKO; GAMA; RIBEIRO, 2020). In contrast to that, unsupervised methodologies rely on the physical laws that determine the state values of the power system and on the constraints imposed by the grid infrastructure in order to perform the training of the models. In particular, (DONON et al., 2020) developed an unsupervised GNN-based architecture to compute the power flow of electrical grids, which is trained to minimize the violation of Kirchhoff's laws at each bus. As shown in the experimental results, this model is much faster than traditional methods (such as the Newton-Raphson algorithm), while also being able to generalize to previously unseen topologies in a zero-shot scenario. In a similar fashion, by making use of concepts from the field of Graph Signal Processing (GSP), (OWERKO; GAMA; RIBEIRO, 2022) devised a GNN that learns the solution to the OPF problem in an unsupervised manner by minimizing the total generation cost directly, while penalizing the violation of the electrical constraints of the grid by means of a custom barrier method. However, none of these studies consider employing a GNN to create a model capable of

solving the OPF problem with the zero-shot property, where, at inference time, it can perform satisfactorily even on grids that had not previously been seen during training.

Hence, in this work, an unsupervised GNN architecture inspired by (DONON et al., 2020) is proposed, which is designed to solve, in a computationally efficient and scalable manner, the OPF in a power grid. The model was conceived to have the capability of adapting to various network topologies and configurations while aiming to minimize the total generation cost of the system.

1.2 Goals

Given the inherent complexity of the OPF problem, the primary goal of this work is to develop a computational framework capable of efficiently solving the OPF in various power grids. Specifically, by leveraging advances in machine learning, with a focus on GNNs, a novel approach for addressing this challenge is presented. The ultimate objective is to create a GNN model that not only minimizes generation costs in a power grid but also significantly outperforms traditional solvers in terms of computation speed. Besides that, the proposed architecture should be able to generalize to topologies not previously seen during training.

1.3 Justification

Traditional approaches to solving OPF problem are often computationally expensive and struggle to scale with the increasing size and complexity of modern power grids. By exploring machine learning-based methodologies, particularly GNNs, it is possible to obtain significant gains in computational speed and efficiency, which would in turn allow for enhanced grid responsiveness, improved system stability, and more effective decision-making in real-time grid operations.

In addition to that, it is important to emphasize that achieving a speedup in the OPF calculation could benefit several applications. For example, this would allow faster response times to transient line outages as well as efficient operation alongside fast switching power devices. Furthermore, it could also expedite a more comprehensive security and risk assessment under uncertainty by Transmission System Operators (TSOs).

1.4 Organization of the Manuscript

The rest of this work is organized as follows. In Chapter 2, the theoretical foundations upon which this work is based are thoroughly explained. Chapter 3 presents the phases this work was divided into and the activities pertaining to each one of them, as

well as a schedule for the completion of these tasks. Chapter 4 specifies the functional and non-functional requirements identified for the proposed GNN model. In Chapter 5, the project and implementation process is described. Afterwards, in Chapter 6, the results obtained from the performed evaluation are presented and discussed. Finally, Chapter 7 gives concluding remarks.

2 Theoretical Foundations

Since GNNs fit well to the graph structure of power grids, using them as a parametrization to the OPF problem is a promising avenue to find solutions in a fast and scalable manner. With that in mind, considering that the goal of this work is to develop a GNN model to compute the OPF of a power grid, it is first important to explain the concepts underlying the functioning of this kind of neural network, as well as provide a mathematical formulation of the OPF problem. In this respect, Section 2.1 describes the main building blocks on which GNNs are based and elucidates the idea of message passing. In Section 2.2, the notation adopted to model a power grid is presented, and the PF and OPF problems are expressed in mathematical terms. The model architecture and the training process detailed in Chapter 3 make use of and build upon the notation and concepts specified in these sections.

2.1 Graph Neural Networks

In the field of machine learning, a Neural Network (NN) is a type of algorithm inspired by the structure and function of the human brain. It consists of layers of interconnected nodes (called neurons) that process information and make predictions. Each neuron takes in a set of inputs, multiplies each input by a weight, and then applies an activation function (usually nonlinear) to the sum of these weighted inputs. The resulting output is passed on to the next layer of neurons, and the process is repeated until the final layer produces the network's prediction.

Training a neural network involves adjusting the weights between neurons to minimize a cost function, which measures the difference between the network's predicted outputs and the actual outputs for a given set of training examples. This is done using an optimization algorithm, typically gradient descent, which iteratively adjusts the values of the function's parameters in the direction of the negative gradient of the function, bringing it closer to one of its minima — which is not necessarily a global minimum, since non-convex functions may have multiple local minima.

GNNs are a class of neural network models designed to operate on graph structures while respecting the topological relationships among the nodes. They were first introduced in (SCARSELLI et al., 2009), having since been further developed and successfully applied to various tasks, such as graph classification (DEFFERRARD; BRESSON; VANDERGHEYNST, 2016), node classification (HAMILTON; YING; LESKOVEC, 2017) and relational reasoning (SANTORO et al., 2017).

In order to understand the principle upon which GNNs are based, a few definitions must first be established. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph, where \mathcal{V} represents the set of N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ corresponds to the set of edges. A vector of input features $\mathbf{x}_i \in \mathbb{R}^D$ is assigned to every node $i \in \mathcal{V}$, where D is the number of the input features. Thus, $\mathbf{X} = [\mathbf{x}_1^T, \dots, \mathbf{x}_N^T] \in \mathbb{R}^{N \times D}$ denotes the input matrix for all nodes in the graph.

Additionally, the topology of the graph is defined by a graph shift operator (GSO). In the context of graph signal processing, a GSO is a mathematical operator that describes the propagation or transformation of signals from a node to its neighbours, capturing the notion of how signals diffuse or spread across the graph. It is typically represented by a matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ (such as the adjacency or the Laplacian matrix), whose entries specify the weight of the connections between pairs of nodes in the underlying graph structure. It should be pointed out that multiplying the GSO by the input matrix can be interpreted as a shift of the inputs across the graph, i.e., the state of each node is updated by a linear combination of the states of the neighbouring nodes, in a form of information exchange among them.

The goal of a GNN model is then to learn a function $\Phi(\mathbf{X}; \mathcal{W}, \mathbf{A})$ that, for a set of parameters \mathcal{W} , takes the underlying graph topology expressed by \mathbf{A} into account and maps, at node-level, the input features \mathbf{X} to a matrix of output features $\mathbf{Y} \in \mathbb{R}^{N \times F}$, where F refers to the number of output features. Therefore, a GNN can be represented by a cascade of L layers, each of which applies a graph convolution operation followed by a nonlinear activation function σ_ℓ :

$$\mathbf{X}^{(\ell)} = \sigma_\ell [\mathbf{A}\mathbf{X}^{(\ell-1)}\mathbf{W}^{(\ell)}], \ell = 1, \dots, L, \quad (2.1)$$

where $\mathbf{X}^{(0)} = \mathbf{X}$ designates the input matrix and $\mathbf{W}^{(\ell)} \in \mathbb{R}^{F_{\ell-1} \times F_\ell}$ is the weight matrix of the ℓ -th layer. At each layer ℓ , the state $\mathbf{X}^{(\ell)} \in \mathbb{R}^{N \times F_\ell}$ contains F_ℓ features and the state of the last layer $\mathbf{X}^{(L)}$ is taken as the output of the GNN, i.e., $\Phi(\mathbf{X}; \mathcal{W}, \mathbf{A}) = \mathbf{X}^{(L)}$.

Figure 1 depicts a simplified block diagram which illustrates the flow of data in a GNN and how the final output of the model is computed.

As a subset of GNN models, the Message Passing Neural Network (MPNN) framework (GILMER et al., 2017) generalizes the implicit information exchange of the graph convolution operation by following an iterative scheme of updating node representations based on the aggregation of neighbour nodes. The message passing paradigm is typically expressed in terms of two types of functions: a message function and an update function.

More specifically, for a given node $i \in \mathcal{V}$, its node features $\mathbf{x}_i^{(\ell-1)}$ at layer $\ell - 1$ are encoded by the message function $\phi^{(\ell)}$ into an M -dimensional message $\mathbf{m}_i^{(\ell)} \in \mathbb{R}^M$:

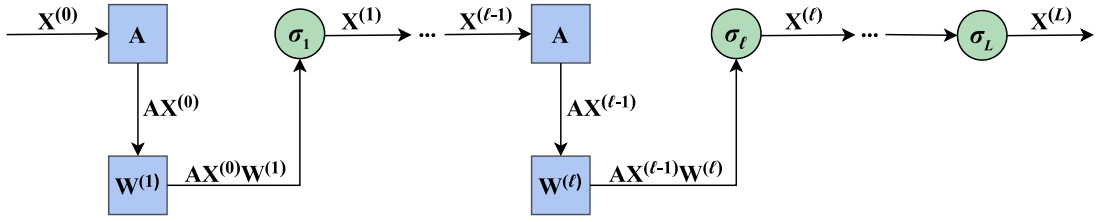


Figure 1 – A Graph Neural Network consists of several layers that take a linear combination of the values of neighbouring nodes $\mathbf{A}\mathbf{X}^{(\ell-1)}$ and weight it by $\mathbf{W}^{(\ell)}$. The output of each layer is then obtained by applying a nonlinearity σ_ℓ to the result of the previous operation.

$$\mathbf{m}_i^{(\ell)} = \bigoplus_{j \in \mathcal{N}(i)} \phi^{(\ell)} \left(\mathbf{x}_i^{(\ell-1)}, \mathbf{x}_j^{(\ell-1)}, \mathbf{e}_{ij} \right), \quad (2.2)$$

where \bigoplus is a permutation invariant aggregation function (e.g., sum, mean or max), $\mathcal{N}(i)$ denotes the neighbours of node i in the graph, and $\mathbf{e}_{ij} \in \mathbb{R}^E$ (vector with E dimensions) symbolizes (optional) edge features from node i to node $j \in \mathcal{N}(i)$.

This message is then used to compute the node features $\mathbf{x}_i^{(\ell)}$ at layer ℓ by means of the update function $\psi^{(\ell)}$:

$$\mathbf{x}_i^{(\ell)} = \psi^{(\ell)} \left(\mathbf{x}_i^{(\ell-1)}, \mathbf{m}_i^{(\ell)} \right). \quad (2.3)$$

Thus, considering a sequence of L layers, the message passing scheme can be represented by the following equation:

$$\mathbf{x}_i^{(\ell)} = \psi^{(\ell)} \left(\mathbf{x}_i^{(\ell-1)}, \bigoplus_{j \in \mathcal{N}(i)} \phi^{(\ell)} \left(\mathbf{x}_i^{(\ell-1)}, \mathbf{x}_j^{(\ell-1)}, \mathbf{e}_{ij} \right) \right), \quad \ell = 1, \dots, L. \quad (2.4)$$

Finally, it must be pointed out that the message functions $\phi^{(\ell)}$ and the update functions $\psi^{(\ell)}$ should be differentiable. They are usually parametrized by neural networks, whose weights can be learned through gradient descent during the training process.

2.2 Power Grid Modelling and the OPF Problem

An electrical grid consists of a set of buses connected by power lines. These elements are characterized by certain physical quantities — referred to as features — that have to be considered when computing the PF and OPF solutions. Most of these quantities are expressed in the per-unit (p.u.) system, which uses the base power and voltage levels of the grid in order to normalize the values. This allows for simpler comparisons of the parameters of distinct grids.

Therefore, each bus can be described by a set of variables and physical properties. The voltage magnitude V and the voltage phase θ are variables that change depending on the state of the grid, while the active power demand P_d , the reactive power demand Q_d , the shunt conductance G_s and the shunt susceptance B_s are physical properties of the bus, which are typically fixed. Besides that, the minimum and the maximum allowed voltage magnitudes, V^{min} and V^{max} , respectively, are determined by regulation.

Similarly, power lines connecting two buses are characterized by several features, such as their resistance r , their reactance x , the total line charging susceptance b , the transformer off nominal turns ratio τ and the transformer phase shift angle θ_{shift} . In addition to that, the admittance of the line can be converted to polar form by applying the transformations $y = \frac{1}{\sqrt{r^2+x^2}}$ and $\delta = \text{atan2}(r, x)$. A line may also have lower and upper limits on the difference in voltage phase between the buses it connects (COFFRIN et al., 2018), which are, respectively, indicated by $\Delta\theta^{min}$ and $\Delta\theta^{max}$.

Furthermore, a bus may have a generator attached to it, which provides active and reactive power to other elements of the grid. The characteristics of a generator are represented by its voltage magnitude set point V_g , its active power output P_g and its reactive power output Q_g . The power produced by each generator is constrained to a certain range by its active power limits, P^{min} and P^{max} , and its reactive power limits, Q^{min} and Q^{max} .

A power grid can be represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, N\}$ is the set of N nodes that correspond to the buses and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges which designate the power lines. Additionally, $\mathcal{V}^G \subseteq \mathcal{V}$ denotes the set of generators in the network.

With that in mind, the steady-solution of the PF problem can be found by solving a system of nonlinear equations given by Kirchhoff's laws. Using polar coordinates, the power balance equations (MILANO, 2010) can be written as:

$$P_i = V_i \sum_{j \in \mathcal{N}(i)} V_j (g_{ij} \cos(\Delta\theta_{ij}) + b_{ij} \sin(\Delta\theta_{ij})), \quad (2.5)$$

$$Q_i = V_i \sum_{j \in \mathcal{N}(i)} V_j (g_{ij} \sin(\Delta\theta_{ij}) - b_{ij} \cos(\Delta\theta_{ij})), \quad (2.6)$$

where P_i and Q_i are the net active and reactive power injections at bus $i \in \mathcal{V}$, $\Delta\theta_{ij}$ is the voltage angle difference between bus i and one of its neighbours $j \in \mathcal{N}(i)$, and g_{ij} and b_{ij} are, respectively, the conductance and the susceptance of the line that connects buses i and j — both of which can be derived from the line admittance.

The goal of power flow balancing is then to determine the voltage magnitudes and phases that satisfy (2.5) and (2.6). Traditionally, this has been done by means of

the Newton-Raphson (NR) method, which works by linearizing the power flow equations through the Jacobian matrix, and then solving the resulting system of linear equations. This iterative process is repeated until a convergence criterion is reached, which typically compares the difference between the estimated and actual power injections at each bus. Nevertheless, this procedure is still computationally expensive for large networks and is not guaranteed to converge, due to the non-convex nature of the problem.

On the other hand, the **Optimal Power Flow** (OPF) problem is a constrained optimization problem that aims to minimize the total generation cost in the grid given the power demanded by the buses, a set of equality constraints (defined by Kirchhoff's laws) and a set of inequality constraints (related to operational restrictions of the equipment). Usually, the objective function to be minimized depends on the active power produced by each generator. Thus, the OPF problem can be expressed through the following equations:

$$\min_{V, \Delta\theta, P_g, Q_g} \sum_{i \in \mathcal{V}^G} c_i(P_{g,i}) \quad (2.7)$$

s. t.

$$P_i = \sum_{j \in \mathcal{V}_i^G} P_{g,j} - \sum_{j \in \mathcal{V}_i^D} P_{d,j}, \quad i \in \mathcal{V}, \quad (2.8)$$

$$Q_i = \sum_{j \in \mathcal{V}_i^G} Q_{g,j} - \sum_{j \in \mathcal{V}_i^D} Q_{d,j}, \quad i \in \mathcal{V}, \quad (2.9)$$

$$V_i^{\min} \leq V_i \leq V_i^{\max}, \quad i \in \mathcal{V}, \quad (2.10)$$

$$\Delta\theta_{ij}^{\min} \leq \Delta\theta_{ij} \leq \Delta\theta_{ij}^{\max}, \quad (i, j) \in \mathcal{E}, \quad (2.11)$$

$$P_{g,i}^{\min} \leq P_{g,i} \leq P_{g,i}^{\max}, \quad i \in \mathcal{V}^G, \quad (2.12)$$

$$Q_{g,i}^{\min} \leq Q_{g,i} \leq Q_{g,i}^{\max}, \quad i \in \mathcal{V}^G, \quad (2.13)$$

where $c_i(P_{g,i})$ is a cost function for generator $i \in \mathcal{V}^G$ (commonly modeled by a quadratic function). In (2.8) and (2.9), the net power injection at bus $i \in \mathcal{V}$ is calculated by subtracting the power consumed by the loads on the bus from the total power injected by generators, with \mathcal{V}_i^G and \mathcal{V}_i^D denoting the sets of generators and loads connected to bus i . The power injections P_i and Q_i must also be equal to the power flowing into and out of the bus, which is determined by the power flow equations (2.5) and (2.6). Moreover, the (complex) power lost due to the shunt admittance $Y_{s,i}$ of the bus is given by Ohm's law, which can be written as $Y_{s,i}(V_i)^2$.

The OPF problem can be solved by means of Interior Point (IP) methods, which are a class of algorithms that can be applied to nonlinear and non-convex optimization problems. These gradient-based methods iteratively minimize an objective function defined as a combination of the cost function to be decreased and a barrier function that penalizes infeasible solutions. At each iteration, the gradient and the Hessian matrix of the objective

function are used to update the decision variables by means of the Newton-Raphson method. This process is repeated until the convergence criteria are met. In the context of power systems, so-called Interior Point Solvers (IPS) are able to yield accurate solutions to the OPF problem. However, in general, they are computationally expensive and slow for large networks ([CASTILLO; O'NEILL, 2013](#)), and have no guarantee of convergence.

3 Methodology

In order to guide and organize the development of this work, it was divided into five distinct work packages, which are described below:

- **WP1 - Preliminary work and literature research:** research on basic concepts of machine learning and optimization (non-convex optimization, neural networks and graph neural networks), literature research on the application of GNNs to power flow problems, and preparation of the software setup and development environment;
- **WP2 - Replication of results of (DONON et al., 2020):** implementation of the training scheme described in the paper, evaluation of the architecture on at least two power grids, and assessment of the zero-shot property claimed by the authors;
- **WP3 - Implementation of a GNN-based algorithm for OPF:** development of a GNN to perform OPF with the zero-shot property, based on the methods presented by (DONON et al., 2020; OWERKO; GAMA; RIBEIRO, 2020) and other papers found in the literature;
- **WP4 - Analysis and results:** evaluation of the proposed architecture against other OPF algorithms, regarding accuracy on the grid whose data was used for the training of the algorithms, accuracy when employing the algorithms on grids other than the one on which the algorithms were trained, and computation time;
- **WP5 - Documentation and presentation:** delivery of a written report explaining the performed work, and preparation of a presentation discussing the most important results achieved.

Taking these work packages into account, the Gantt diagram shown in Figure 2 depicts the planned time periods allotted to each activity. As can be observed, the work packages WP1 and WP2 should be carried out simultaneously, lasting for about a month. After the completion of these tasks, the actual implementation of the GNN-based algorithm for solving the OPF problem, as specified in WP3, should begin, with an estimated duration of two months. Approximately two weeks before the finalization of WP3, the phase of evaluating the developed model on various power grid topologies and analysing the obtained results (e.g., through the generation of plots), as described by WP4, should commence, with a scheduled completion time of 20 days. Lastly, the process of producing the written report of the performed work, as defined by WP5, should start around 20 days after the beginning of the implementation phase (in order to capture the

most important aspects of this step), being concluded at the end of this project, after approximately 45 days.

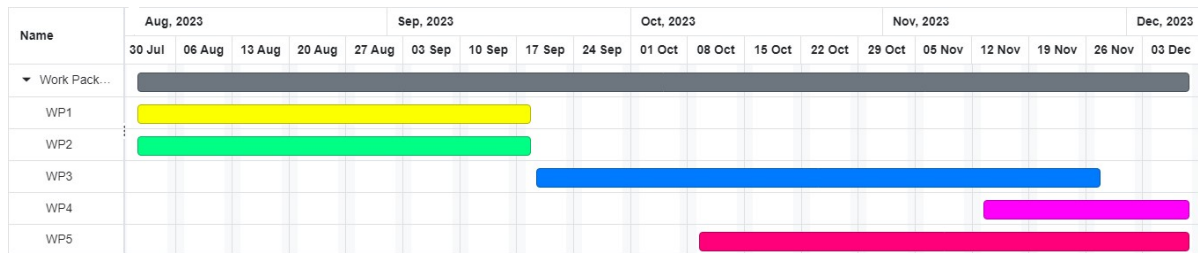


Figure 2 – The time slots allotted to each work package are just an estimation of the expected duration of each task based on their assessed complexity. The actual completion time of each work package may differ slightly according to the progress made and to the difficulties encountered at each step of the project.

4 Requirements Specification

In this chapter, the functional and non-functional requirements specified for the GNN model to be developed are presented and explained. It should be noted that some of the listed requirements may be more relevant to possible future extensions of this work.

With that in mind, the following functional requirements were identified:

- **Optimization accuracy:** the GNN model should accurately solve the OPF problem, ensuring that it minimizes generation costs while satisfying operational constraints and demand requirements;
- **Speed and efficiency:** the primary functional requirement is that the GNN model performs the OPF calculations significantly faster than traditional solvers, enabling real-time or near-real-time operation of the power grid;
- **Scalability:** the model should be capable of handling power grids of varying sizes, from small distribution networks to large-scale transmission systems with thousands of buses, while maintaining its computational efficiency;
- **Integration with grid data:** the model should be able to accept input data representing the topology, parameters, and operational constraints of the power grid. The data include information about buses, generators, loads, transmission lines, and other components.

Furthermore, the non-functional requirements described below should be taken into account in the evaluation of the usefulness and adequateness of the proposed model.

- **Generalization capability:** the GNN model should generalize well to different grid topologies, configurations, and sizes. It should be able to handle diverse network layouts without requiring retraining;
- **Model robustness and stability:** the GNN model should be stable and robust, providing reliable solutions even in the presence of noisy grid data;
- **Resource efficiency:** the model should utilize computational resources efficiently, enabling it to be deployed on diverse hardware platforms;
- **Memory efficiency:** the model should manage memory usage effectively to avoid excessive memory overhead, enabling it to handle large-scale power grids without causing memory-related issues (such as memory fragmentation or out-of-memory problems);

- **Maintainability and extensibility:** the GNN model should be designed in a modular and well-structured manner, facilitating easy maintenance, updates, and extensions as new features or improvements are introduced;

5 Development

This chapter describes the development process of the GNN model proposed to solve the OPF problem. In Section 5.1, the main technologies used for the implementation are listed. Section 5.2 details the actual project and implementation decisions with regards to the model, from its architecture to the training procedure.

5.1 Used Technologies

This project was developed entirely in Python 3.10, with the PyTorch (PASZKE et al., 2019) (version 1.13) and PyTorch Geometric (FEY; LENSSEN, 2019) (version 2.0) libraries being used to create the train and test sets, as well as to define, train and evaluate the GNN models utilized to solve the OPF problem. Apart from that, the Numpy (HARRIS et al., 2020) and Matplotlib (HUNTER, 2007) libraries were employed to store the evaluation results and to generate the plots shown in Section 6.3.

5.2 Project and Implementation

In (DONON et al., 2020), a message passing GNN is proposed to compute the power flow of an electrical grid. The architecture of the model, which is called Graph Neural Solver (GNS) by its authors, is based on an iterative process that applies "correction" updates to the variables of each bus (voltage magnitude, voltage phase and latent message). The model is trained in an unsupervised manner to directly minimize the violation of Kirchhoff's laws at each bus — instead of minimizing the distance between its predictions and the output of a particular solver (HANSEN; ANFINSEN; BIANCHI, 2023; DONON et al., 2019). One main advantage of the unsupervised method is that this architecture displays a form of zero-shot property, i.e., a model trained on a specific power grid is able to achieve, during inference time, decent performance on grids that had not been previously seen during training, regardless of their topology.

In this section, a new methodology for OPF calculation based on the structure of the GNS is presented in Subsection 5.2.1. For example, the inequality constraints (2.10)-(2.13) had to be taken into account in the new architecture. Furthermore, Subsection 5.2.2 details the procedure adopted to train the model.

5.2.1 Model Architecture

The proposed architecture utilizes a GNN to infer the state of the grid, i.e., the voltage magnitude and phase at each bus, along with the optimal set point of the generators' active and reactive power — whereas (DONON et al., 2020) only focus on finding the steady-state voltage magnitude and phase values in the grid. The model employs an iterative scheme in which these variables are progressively updated based on the values of the previous iteration. Additionally, the algorithm makes use of a message passing mechanism, where d -dimensional messages are exchanged between buses. These messages can be viewed as a way to encode the state of a bus and information about its neighbours into an abstract space (DONON et al., 2020).

Figure 3 depicts a block diagram representation of the proposed architecture, with arrows indicating the flow of data between its elements. The following subsections provide details on each of the architectural components.

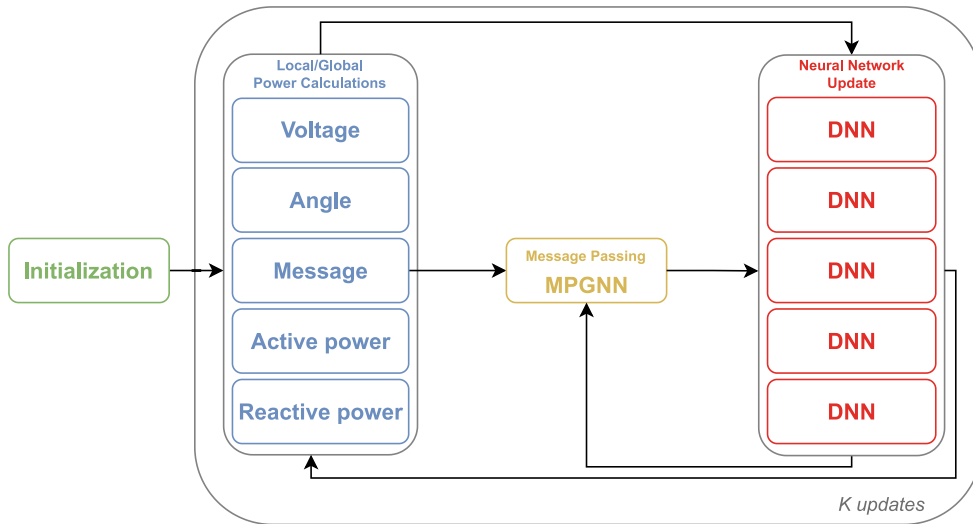


Figure 3 – After the initialization step, the model variables are used to perform the power flow computation. The results of these calculations, along with the messages produced by the message-passing GNN, are then passed as inputs to DNNs that generate the updates to each of the variables and to the MPGNN. This process is repeated for K iterations.

5.2.1.1 Initialization

The voltage magnitudes of all buses are set to 1 p.u., except for the ones connected to a generator, for which the voltage magnitude is the voltage set point of the generator. All voltage phase angles are initialized to 0 radians, and the GNN messages at each buses are also set to zero.

A DC-OPF solver is run on the grid in order to set the initial active power output of the generators. This was done due to the fact that DC-OPF has a low computational cost

and is often used as a warm start to the OPF problem. With regards to the initialization of the reactive power output of the generators, the following procedure was adopted: firstly, the ratio between the total reactive power that can be produced by the generators and the total reactive demand across the entire network is calculated. The initial reactive power of the each generators is then set by multiplying this ratio by the corresponding maximum allowed reactive power output.

5.2.1.2 Global Power Consumption Calculation

After the initialization, for each update $k = 0, \dots, K - 1$, by summing the power consumed by every bus in the grid, it is possible to compute the global active and reactive power consumption — designated respectively by P_{global} and Q_{global} — based on the voltage magnitude and phase values for each bus, the power line characteristics and the demands and shunt loads of the buses. The active power consumption must also include the losses due to Joule dissipation (P_{joule}). The calculation of the global power consumption is thus given by (5.1)-(5.3):

$$P_{global}^{(k)} = \left(\sum_{i \in \mathcal{V}} P_{d,i} + G_{s,i} (V_i^{(k)})^2 \right) + P_{joule}^{(k)}, \quad (5.1)$$

$$P_{joule}^{(k)} = \sum_{(i,j) \in \mathcal{E}} \left| V_i^{(k)} V_j^{(k)} y_{ij} \frac{1}{\tau_{ij}} \left(\sin(\theta_i^{(k)} - \theta_j^{(k)} - \delta_{ij} - \theta_{shift,ij}) + \sin(\theta_j^{(k)} - \theta_i^{(k)} - \delta_{ij} + \theta_{shift,ij}) \right) + \left(\frac{V_i^{(k)}}{\tau_{ij}} \right)^2 y_{ij} \sin(\delta_{ij}) + (V_j^{(k)})^2 y_{ij} \sin(\delta_{ij}) \right|, \quad (5.2)$$

$$Q_{global}^{(k)} = \left(\sum_{i \in \mathcal{V}} Q_{d,i} - B_{s,i} (V_i^{(k)})^2 \right). \quad (5.3)$$

5.2.1.3 Local Power Imbalance Calculation

Furthermore, the local active and reactive power imbalance given by ΔP and ΔQ , respectively, can be calculated for each bus at update k through (5.4) and (5.5). These equations can be seen as a measure of the violation of Kirchhoff's laws, which, in simple terms, state that the power flowing into a bus should be equal to the power flowing out of that bus. It should be noted that, in the equations below, power lines connecting two buses are denoted by their "from" and "to" sides, which are represented by the sets \mathcal{U} and \mathcal{D} , respectively, and indicate the direction of the energy flow:

$$\begin{aligned}
\Delta P_i^{(k)} &= \left(P_{g,i}^{(k)} - P_{d,i}^{(k)} - G_{s,i}^{(k)} \left(V_i^{(k)} \right)^2 \right) \\
&+ \sum_{\substack{i \in \mathcal{U} \\ j \in \mathcal{N}(i)}} \left[V_i^{(k)} V_j^{(k)} y_{ij} \frac{1}{\tau_{ij}} \sin(\theta_i^{(k)} - \theta_j^{(k)} - \delta_{ij} - \theta_{shift,ij}) + \left(\frac{V_i^{(k)}}{\tau_{ij}} \right)^2 y_{ij} \sin(\delta_{ij}) \right] \\
&+ \sum_{\substack{i \in \mathcal{D} \\ j \in \mathcal{N}(i)}} \left[V_i^{(k)} V_j^{(k)} y_{ij} \frac{1}{\tau_{ij}} \sin(\theta_i^{(k)} - \theta_j^{(k)} - \delta_{ij} + \theta_{shift,ij}) + \left(V_i^{(k)} \right)^2 y_{ij} \sin(\delta_{ij}) \right],
\end{aligned} \tag{5.4}$$

$$\begin{aligned}
\Delta Q_i^{(k)} &= \left(Q_{g,i}^{(k)} - Q_{d,i}^{(k)} + B_{s,i}^{(k)} \left(V_i^{(k)} \right)^2 \right) \\
&+ \sum_{\substack{i \in \mathcal{U} \\ j \in \mathcal{N}(i)}} \left[-V_i^{(k)} V_j^{(k)} y_{ij} \frac{1}{\tau_{ij}} \cos(\theta_i^{(k)} - \theta_j^{(k)} - \delta_{ij} - \theta_{shift,ij}) + \left(\frac{V_i^{(k)}}{\tau_{ij}} \right)^2 \left(y_{ij} \cos(\delta_{ij}) - \frac{b_{ij}}{2} \right) \right] \\
&+ \sum_{\substack{i \in \mathcal{D} \\ j \in \mathcal{N}(i)}} \left[-V_i^{(k)} V_j^{(k)} y_{ij} \frac{1}{\tau_{ij}} \cos(\theta_i^{(k)} - \theta_j^{(k)} - \delta_{ij} + \theta_{shift,ij}) + \left(V_i^{(k)} \right)^2 \left(y_{ij} \sin(\delta_{ij}) - \frac{b_{ij}}{2} \right) \right].
\end{aligned} \tag{5.5}$$

5.2.1.4 Message Passing Mechanism

Besides the attributes that relate to physical properties of the buses, a message variable is attributed to each bus in the grid. These messages carry no direct physical meaning, but their propagation through the network has been shown to be a suitable way to compute the power flow (DONON et al., 2019).

The message function ϕ is performed by a deep neural network, which takes as input a vector $\mathbf{m}^{(k)} = [m_1^{(k)}, \dots, m_N^{(k)}]$ containing the messages associated with every bus, as well as the physical characteristics of all the power lines in the grid, represented by $\mathbf{l} = (\mathbf{y}, \boldsymbol{\delta}, \mathbf{b}, \boldsymbol{\tau}, \boldsymbol{\theta}_{shift})$. The output of this neural network is a vector of processed messages $\overline{\mathbf{m}}^{(k)}$ with entries for each bus:

$$\overline{\mathbf{m}}^{(k)} = \phi \left(\mathbf{m}^{(k)}, \mathbf{l} \right). \tag{5.6}$$

By multiplying the vector $\overline{\mathbf{m}}^{(k)}$ by the incidence matrix \mathbf{A} , that indicates which buses are connected by a power line, it is then possible to obtain a vector $\mathbf{z}^{(k)}$ that stores, for each bus, the sum of the processed messages of all its neighbours. This aggregation operation can be interpreted as the propagation of messages among neighbouring buses:

$$\mathbf{z}^{(k)} = \mathbf{A} \left(\overline{\mathbf{m}}^{(k)} \right)^T. \tag{5.7}$$

5.2.1.5 Neural Network Update

The rules used to update the voltage magnitudes and phases of each bus, the messages, and the active and reactive power outputs of the generators are described by (5.8)-(5.12). The term $\mathbf{s}^{(k)} = (\mathbf{V}^{(k)}, \boldsymbol{\theta}^{(k)}, \mathbf{m}^{(k)}, \Delta \mathbf{P}^{(k)}, \Delta \mathbf{Q}^{(k)})$ contains the values of the voltage magnitude and phase, the message, and the local active and reactive power imbalance for every bus in the grid at update k , while $\mathbf{z}^{(k)}$ is given by (5.7). The terms $\mathbf{s}_g^{(k)}$ and $\mathbf{z}_g^{(k)}$ are defined similarly to their counterparts, but only include information about buses that are attached to generators:

$$V_i^{(k+1)} = \begin{cases} V_{g,i}, & \text{if } i \in \mathcal{V}^G \\ V_i^{(k)} + L_{v,i}^{(k)}(\mathbf{s}^{(k)}, \mathbf{z}^{(k)}), & \text{otherwise} \end{cases}, \quad (5.8)$$

$$\theta_i^{(k+1)} = \theta_i^{(k)} + L_{\theta,i}^{(k)}(\mathbf{s}^{(k)}, \mathbf{z}^{(k)}), \quad i \in \mathcal{V}, \quad (5.9)$$

$$m_i^{(k+1)} = m_i^{(k)} + L_{m,i}^{(k)}(\mathbf{s}^{(k)}, \mathbf{z}^{(k)}), \quad i \in \mathcal{V}, \quad (5.10)$$

$$P_{g,i}^{(k+1)} = P_{g,i}^{(k)} + L_{p,i}^{(k)}(\mathbf{s}_g^{(k)}, \mathbf{z}_g^{(k)}), \quad i \in \mathcal{V}^G, \quad (5.11)$$

$$Q_{g,i}^{(k+1)} = Q_{g,i}^{(k)} + L_{q,i}^{(k)}(\mathbf{s}_g^{(k)}, \mathbf{z}_g^{(k)}), \quad i \in \mathcal{V}^G. \quad (5.12)$$

The functions $L_v^{(k)}$, $L_\theta^{(k)}$, $L_m^{(k)}$, $L_p^{(k)}$ and $L_q^{(k)}$ are deep neural networks which are specific to an iteration k and are trained to perform updates to the model variables based on the state of the grid and on the sum of messages exchanged between neighbouring nodes. The parameters of these neural networks, as well as those of the message function (5.6), are learned during training through gradient descent. In the equations above, the subscript i was utilized to indicate the i -th element of the vector returned as the output of these functions. Moreover, as defined in (5.8), the voltage magnitude of the buses connected to a generator remains fixed at the voltage set point of their respective generator. It is important to emphasize that, in contrast to (DONON et al., 2020), who use a custom compensation rule to control the active power output of each generator (in order to ensure a global active equilibrium) and do not even consider the global reactive power balance in the grid, the proposed algorithm makes use of neural networks with learnable parameters to calculate the active and reactive power outputs of the generators.

It should be noted that these functions could be realized through other approaches. For instance, considering that the the updates to the model variables take the values of the previous iterations into account, one valid alternative would be the utilization

of Long Short-Term Memory (LSTM) modules to perform the update steps. LSTMs (HOCHREITER; SCHMIDHUBER, 1997) are a type of recurrent neural network (RNN) architecture designed to be able to capture long-range dependencies and relationships between sequential data effectively. In order to do that, they make use of so-called gates, which allow them to selectively update, retain, or discard information at each time step. However, after preliminary tests, it was verified that using LSTMs instead of regular DNNs lead to significantly longer training and inference times - which can most likely be attributed to the sequential nature of LSTMs. Thus, in the proposed architecture, deep neural networks are used to perform the update functions.

5.2.2 Training

In regards to the training process, the learnable weights of the GNN model are optimized against a multi-objective function that takes into account the local power balance at each bus, the constraints defined by (2.8)-(2.13) and the total generation cost in the grid. Given a batch of T samples, at the end of each update k , the violation of Kirchhoff's laws ((5.4) and (5.5)) is computed for each bus of every sample $t = 1, \dots, T$. The loss function ℓ used to minimize the local power imbalance is then defined by the following weighted average:

$$\ell = \frac{1}{T} \sum_{t=1}^T \sum_{k=1}^K \frac{\gamma^{\frac{K-k}{\eta}}}{N} \sum_{i=1}^N \left[(\Delta P_{i,t}^{(k)})^2 + (\Delta Q_{i,t}^{(k)})^2 \right], \quad (5.13)$$

where the discount factor $0 < \gamma < 1$ and the weighting factor $\eta \geq 1$ are hyperparameters of the model. The discount factor gives more weight to the contribution of later iterations, while the weighting factor can be adjusted to further control the magnitude of the local loss for each update.

As discussed in Chapter 2, besides the generation cost function that is to be minimized (2.7), the OPF problem consists of a set of equality constraints (2.8)-(2.9) related to the flow of active and reactive power in the grid and of several inequality constraints (2.10)-(2.13) that establish limits on the range of values the voltage magnitude and phase of the buses, as well as the active and reactive power outputs of the generators can assume. Hence, considering the complexity of solving the OPF directly, due to its non-convex nature, its constraints can instead be approximated by means of penalty functions, which quantify the degree to which the constraints are violated. By minimizing the penalized objective function, approximate solutions that satisfy the constraints to some extent can thus be computed. This leads to an unconstrained minimization problem that can be solved, e.g., through gradient descent, if the chosen penalty functions are differentiable. Nevertheless, it should be noted that the parametrization obtained through this approach may result in outputs that are infeasible for the constrained problem.

Therefore, in order to ensure a global power equilibrium in the grid, i.e., that the power produced by the generators satisfies the power demanded by the buses, the mean squared error was used as an equality penalty function for both the active and the reactive power, being denoted by P_{eq} and Q_{eq} , respectively. This constraint is applied to every update, leading to the loss functions below:

$$P_{eq} = \frac{1}{T} \sum_{t=1}^T \sum_{k=1}^K \left[\left(\sum_{i \in \mathcal{V}^G} P_{g,i,t}^{(k)} \right) - P_{global,t}^{(k)} \right]^2, \quad (5.14)$$

$$Q_{eq} = \frac{1}{T} \sum_{t=1}^T \sum_{k=1}^K \left[\left(\sum_{i \in \mathcal{V}^G} Q_{g,i,t}^{(k)} \right) - Q_{global,t}^{(k)} \right]^2. \quad (5.15)$$

For the inequality constraints (2.10)-(2.13), the extended log-barrier function introduced in (OWERKO; GAMA; RIBEIRO, 2022) was chosen as a penalty function. This function makes use of the extended logarithm $\overline{\log}_\zeta$, which is defined as:

$$\overline{\log}_\zeta(u) := \begin{cases} \log(u), & \text{if } u \geq 1/\zeta \\ \zeta(u + \frac{1}{\zeta}) - \log\left(\frac{1}{\zeta}\right), & \text{otherwise} \end{cases}, \quad (5.16)$$

The parameter $\zeta \geq 0$ determines the maximum value of the function's derivative. The extended log-barrier function is then defined as:

$$\mu_\xi(u) := -(1/\xi)\overline{\log}_\zeta(-u), \quad (5.17)$$

where $\xi > 0$ is a parameter, with larger values of ξ providing better approximations to the indicator function. Unlike the traditional log-barrier method (BOYD; VANDENBERGHE, 2004), the extended log-barrier function is defined on \mathbb{R}^+ and can be applied to infeasible values, i.e., values that violate the constraints — whereas the classical logarithm's output is undefined for values outside the feasibility set. This is important since the model parameters are randomly initialized and, at first, may not produce feasible solutions.

The total generation cost in the grid depends on the active power output and on the unit costs of each generator. Therefore, the cost function can be defined as the average generation cost across all samples for the last update K :

$$C(P_g^{(K)}) = \frac{1}{T} \sum_{t=1}^T \sum_{i \in \mathcal{V}^G} c_{i,t} (P_{g,i,t}^{(K)}). \quad (5.18)$$

Finally, the complete objective function, which is to be minimized by means of gradient descent, is given below:

$$\begin{aligned}
\mathcal{L} &= \lambda_1 \ell + \lambda_2 P_{eq} + \lambda_3 Q_{eq} \\
&+ \lambda_4 \sum_{t=1}^T \sum_{i \in \mathcal{V}} \mu_\xi \left(V_{i,t}^{min} - V_{i,t}^{(K)} \right) + \lambda_5 \sum_{t=1}^T \sum_{i \in \mathcal{V}} \mu_\xi \left(V_{i,t}^{(K)} - V_{i,t}^{max} \right) \\
&+ \lambda_6 \sum_{t=1}^T \sum_{(i,j) \in \mathcal{E}} \mu_\xi \left(\Delta \theta_{ij,t}^{min} - \Delta \theta_{ij,t}^{(K)} \right) + \lambda_7 \sum_{t=1}^T \sum_{(i,j) \in \mathcal{E}} \mu_\xi \left(\Delta \theta_{ij,t}^{(K)} - \Delta \theta_{ij,t}^{max} \right) \\
&+ \lambda_8 \sum_{t=1}^T \sum_{i \in \mathcal{V}^g} \mu_\xi \left(P_{g,i,t}^{min} - P_{g,i,t}^{(K)} \right) + \lambda_9 \sum_{t=1}^T \sum_{i \in \mathcal{V}^g} \mu_\xi \left(P_{g,i,t}^{(K)} - P_{g,i,t}^{max} \right) \\
&+ \lambda_{10} \sum_{t=1}^T \sum_{i \in \mathcal{V}^g} \mu_\xi \left(Q_{g,i,t}^{min} - Q_{g,i,t}^{(K)} \right) + \lambda_{11} \sum_{t=1}^T \sum_{i \in \mathcal{V}^g} \mu_\xi \left(Q_{g,i,t}^{(K)} - Q_{g,i,t}^{max} \right) \\
&+ \lambda_{12} C \left(P_g^{(K)} \right).
\end{aligned} \tag{5.19}$$

The weight parameters $\lambda_i \geq 0, i = 1, \dots, 12$, can be adjusted so as to strike a balance between the optimality and the feasibility of the solutions. For instance, by increasing the value of the parameter λ_{12} , it would be possible to achieve results with lower generation costs, but that could lead to more constraint violations. On the other hand, adding more weight to the terms related to the constraints could trade off a reduction in the number and severity of the violations for less optimal solutions.

6 Evaluation

This chapter discusses the results obtained from evaluating different trained instances of the proposed model on various power grids. Section 6.1 explains the methodology applied to the generation of the used datasets. In Section 6.2, the different metrics against which the model was evaluated are presented. Finally, Section 6.3 examines the actual performance of the model with regards to the considered evaluation metrics.

6.1 Dataset Generation

In order to construct the datasets utilized both for training and evaluation purposes, a methodology similar to the approaches used in (DONON et al., 2020) and (OWERKO; GAMA; RIBEIRO, 2022) was followed. In this sense, for a particular power grid, the values of the parameters r , x and b of the power lines, the active and reactive power demands of each bus, as well as the cost coefficients used to calculate the unit cost of every generator, are sampled from a uniform distribution around 90% and 110% of their reference values. The voltage set point of the generators are sampled uniformly between 0.95 p.u. and 1.05 p.u., and their initial reactive power outputs are obtained by sampling from a uniform distribution around 95% and 105% of the values obtained after performing the initialization step described in Subsection 5.2.1.1. Therefore, it is possible to generate various samples that correspond to distinct power grid states, helping improve the generalization capability of models trained on that dataset.

6.2 Evaluation Metrics

With respect to the metrics employed to evaluate the proposed architecture, it is important that they relate to the requirements specified in Chapter 4. With that in mind, the metrics chosen to accomplish this were the computation time taken by the GNN model to calculate the OPF in a power grid as well as the generation cost of the produced solutions.

Hence, by comparing the model to an IP solver with regards to these metrics, it is possible to determine if the GNN-based approach is able to adequately minimize generation costs and to compute solutions to the OPF problem faster than traditional methods. In addition to that, analyzing the computation time of the model across various grid sizes allows for the verification of its scalability. Moreover, its generalization capability can be evaluated by checking if an instance of the model can achieve good performance when tested on grids not seen during training.

Lastly, it is relevant to emphasize that, although these metrics alone do not capture all of the non-functional requirements listed in Chapter 4, the model was carefully designed so as to ensure robustness and efficient use of computational resources (especially memory), and the code was developed in a way that facilitates its maintenance.

6.3 Evaluation Results

This section discusses the evaluation results obtained after training the proposed algorithm on different power grids — all of which are available in MATPOWER (ZIMMERMAN; MURILLO-SÁNCHEZ; THOMAS, 2011). In Subsection 6.3.1, the consistency of the model’s solutions are validated using the IP method as a ground-truth. For that, separate models were trained on four different cases (*case9*, *case14*, *case39* and *case118*) and their outputs were compared to those of the IPS with regards to active power flow. In Subsection 6.3.2, both methods are compared in terms of total generation cost and computation time, with a model trained on multiple cases being used for the evaluation. It is shown that the GNN is able to produce results with lower generation costs than an interior point solver, while being much faster than the latter. Furthermore, the generalization capability of the architecture is examined by training a model on a single case (*case300*) and evaluating its performance (also based on generation cost and computation time) on previously unseen grids. Lastly, Subsection 6.3.3 analyzes the computational scalability of the proposed method for different grid sizes, ranging from fewer than ten to over 2000 buses. The achieved results demonstrate that the GNN model (trained on *case2383wp*) scales much better than a traditional OPF solver as the number of buses increases — considering that the computation time of the former stays basically constant regardless of the size of the grid.

Table 1 shows the most relevant hyperparameters of the models trained on only one of the considered power grids: *case9*, *case14*, *case39*, *case118*, *case300* and *case2383wp*. These values were selected empirically after an analysis of the performance of the models on the test sets. The training datasets of each grid consisted of 1000 samples generated according to the methodology described in Section 6.1.

Case	Epochs	Batch size	Correction updates
<i>case9</i>	200	512	10
<i>case14</i>	150	512	10
<i>case39</i>	100	512	15
<i>case118</i>	150	512	10
<i>case300</i>	100	128	10
<i>case2383wp</i>	100	128	10

Table 1 – Selected hyperparameters of the best-performing models trained on a single power grid.

Apart from that, another model was trained with 2500 samples distributed evenly across multiple power grids (*case9*, *case14*, *case39*, *case118* and *case300*). Taking the higher amount of training samples into consideration, this model was trained for 50 epochs with a batch size of 128.

Additionally, for every trained model, the dimension of the latent messages was set to 10, the number of hidden layers used in the neural networks was set to 2 and leaky ReLU was chosen as the activation function of the neural networks. Besides that, a discount factor of $\gamma = 0.5$ was used for the loss function related to the local power imbalance of the buses.

For the models trained exclusively on *case9*, *case14*, *case39* or *case118*, the Adam optimizer with a learning rate of 3×10^{-3} was utilized for all parameters except for the ones related to the neural networks responsible for updating voltage and phase values, for which a learning rate of 1×10^{-3} was used. For the GNN instances only trained on *case300* or *case2383wp*, given the larger number of buses in these grids and the wider range of values that can be assumed by the model variables, in order to avoid instability during training, the learning rate was reduced to 3×10^{-4} for all parameters, with the exception of the ones associated with the neural network that performs the message function. For the update of these parameters, a learning rate of 6×10^{-4} was used. Lastly, for the model trained on multiples cases, the parameters corresponding to the update of the voltage magnitude and phase of the buses had a learning rate of 3×10^{-4} , while the learning rate of the neural networks which update the power outputs of the generators was set to 6×10^{-4} , and the learning rate of the message passing module was defined as 1×10^{-3} .

In all cases, the exponential decay rates of the first and second moments were set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$, respectively. Gradient clipping was also applied, with a maximum clip value of 1×10^{-3} . All models were trained and evaluated on an NVIDIA GTX 1080Ti GPU with 11GB of VRAM, except for the one trained on multiple cases, for which a Tesla T4 GPU with 16GB was utilized.

6.3.1 Model Validation

In this and in the following subsections, the default interior point solver from PYPOWER ([LINCOLN, 2011](#)) (which is a port of MATPOWER to Python) was utilized for all comparisons with the proposed algorithm. Firstly, as a way to check the validity of the outputs produced by the GNN, Figure 4 displays, for a random test sample, a juxtaposition between the solution of a model trained on *case9* and the one of the IPS in terms of the generators' active power output and of the active power flowing through the lines connecting the buses.

As illustrated by this plot, the GNN generates results that closely resemble those of

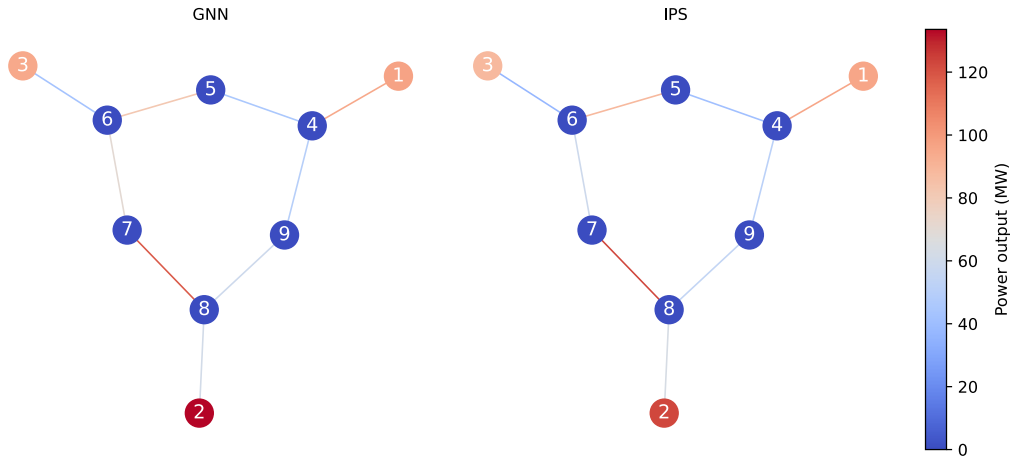


Figure 4 – Comparison of the active power output of the generators (nodes 1-3) and of the active power on the lines when solving the OPF for *case9* using the GNN model (left graph) and the IPS (right graph). The model was trained on *case9*.

the IP method, considering that the mean and maximum deviations of the active set points of the generators with respect to the IPS were 5.57% and 8.76%, respectively. In the case of the active power flow on the lines, the mean and max deviations were, respectively, 8.15% and 18.81% — with most lines having a deviation of less than 10%. This suggests that the model indeed learns how to compute the OPF of a grid in a meaningful way.

Moreover, Figure 5 shows the total difference of the GNN model when compared to the IPS with respect to the active flow on the power lines. In this experiment, we compare the solutions of the GNN when tested with data from the same grid on which the model was trained. Similarly to (DONON et al., 2020), only the 50% largest active flows (in absolute value) were kept for the calculation of the relative error, since the percentage of error can explode for small power flows. As can be observed, the models are able to achieve results that are reasonably close to the outputs of the IPS, indicating that the proposed architecture can indeed be used as a valid way of solving the OPF problem — also considering that, since the problem is non-convex, the IP method does not necessarily compute optimal solutions. It should be pointed out that the test sets only contain samples that converged with the IP method.

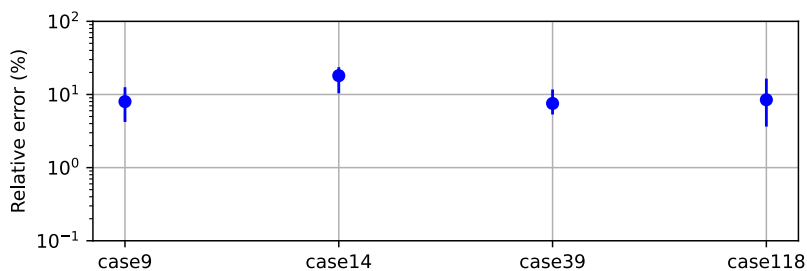


Figure 5 – Relative deviation (20th percentile, median, 80th percentile) of the proposed model with regards to the IPS in terms of active flow of the power lines.

6.3.2 Evaluation on Standard IEEE Power Grids

In this subsection, we compare the cost and speed of the proposed algorithm to the solutions provided by the IP method. To this end, two models were used: one of them was trained on data from multiple power grids (*case9*, *case14*, *case39*, *case118* and *case300*), while the other was trained on a dataset based on *case300* (using the hyperparameters described at the beginning of Section 6.3). The latter was evaluated on various other power grids not seen during training — as a way to verify the zero-shot property.

Furthermore, given that the models may occasionally produce outputs which violate the maximum active power limits of the generators, the excess power was redistributed across the generators of the grid according to the merit order principle, i.e., the excess power is first distributed to the generator with the cheapest generation cost that still has not expended all of its active generation capacity, then to the second cheapest generator and so on. In the case of the reactive power, violations are redistributed proportionally to the remaining reactive capacity of the generators. The application of these techniques ensures that no power constraints are violated. Also, it should be noted that no voltage magnitude and phase violations occurred for any of the test samples.

Figures 6 and 7 depict, respectively, the relative generation cost and the computation time of the GNN model trained on multiple cases and evaluated on each one of them, when compared to the IPS. In Figure 6, it can be seen that the model outputs solutions with slightly lower generation costs on average for all test cases. Besides that, Figure 7 shows that the GNN is much faster (at least by one order of magnitude) and scales much better than the IP method: while the average computation time of the GNN remained basically constant, the computation time of the IPS becomes longer as the number of buses increases.

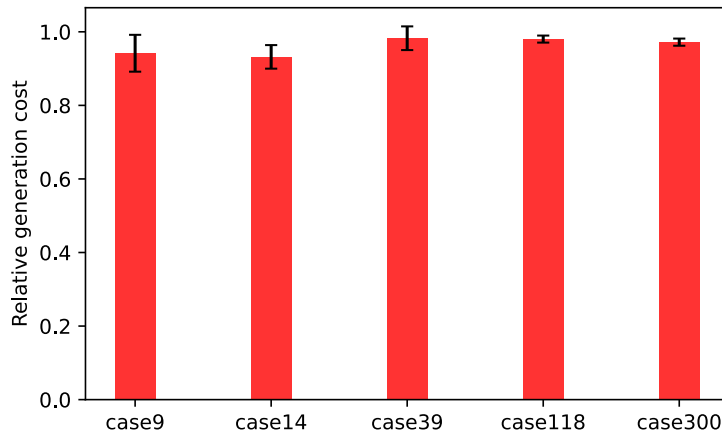


Figure 6 – Relative generation cost of a GNN model with regards to the IPS, when trained on multiple cases and tested on each one of them. In the smaller grids, since each grid element has a stronger contribution to the final result, the noise added to the features of the test samples leads to a higher variance.

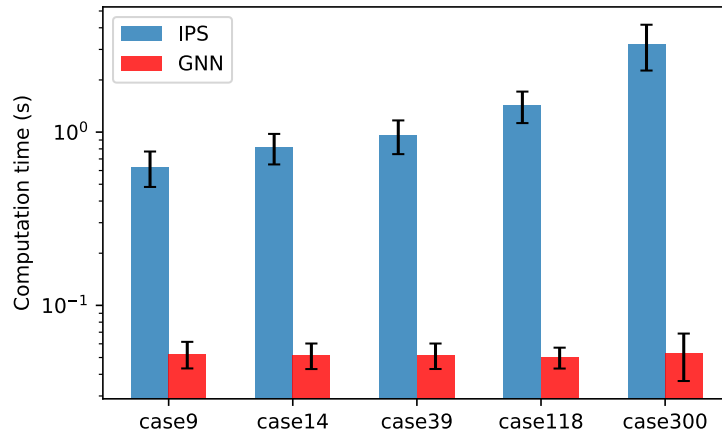


Figure 7 – Comparison of the computation time of the GNN model (red bars) and the IPS (blue bars), when trained on *case9*, *case14*, *case39*, *case118* and *case300* and tested on each of these power grids.

On the other hand, Figure 8 showcases the generalization capability of the proposed architecture, considering that the model which was only trained on *case300* can achieve good performance even on grids that had not been seen during training (zero-shot property). It is also important to highlight that, for *case118* and *case300*, the relative generation costs of this model are quite close to those of the previous one — although it must be noted that this instance of the GNN calculated solutions with marginally higher generation costs for the smaller grids. Similarly to the results presented before, computation time was significantly improved in relation to the IPS, as shown in Figure 9. For *case9*, due to the small size of this grid, the values attributed to the properties of each grid element have a larger impact on the calculations performed by the model. Thus, the noise injected in the test samples causes a high variance in the computation time.

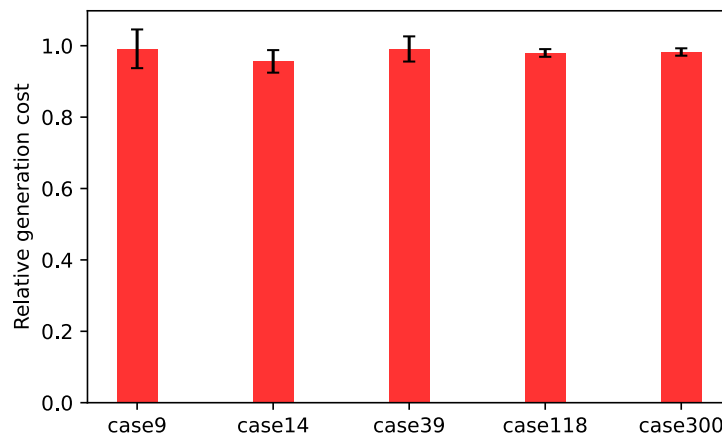


Figure 8 – Relative generation cost of a GNN model with regards to the IPS, when trained on *case300* and tested on other power grids.

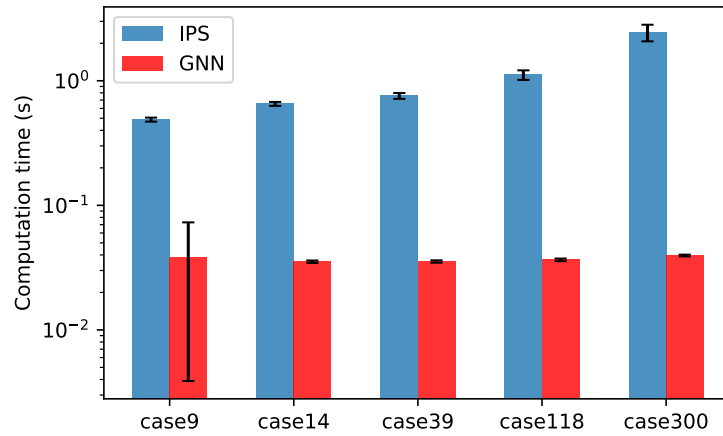


Figure 9 – Comparison of the computation time of the GNN model (red bars) and the IPS (blue bars), when trained on *case300* and tested on other power grids.

6.3.3 Scalability Evaluation

In addition to the analyses of the previous subsections, we validate whether this architecture is applicable to much larger and more complex networks (especially with respect to computation time). To this end, a model was trained on *case2383wp*, which represents the Polish power system during winter 1999-2000 peak conditions. To the best of the author’s knowledge, there is no other work that solves the AC variant of the OPF problem beyond the 118-bus system test case.

As can be seen in Figure 10, the GNN model is able to generate solutions whose cost is reasonably close to that of a traditional IPS. In particular, for *case2383wp*, the generation cost was on average approximately 5% higher when compared to the IP method. This indicates that, with respect to the cost metric, the proposed model is able to achieve satisfactory performance even in very large networks.

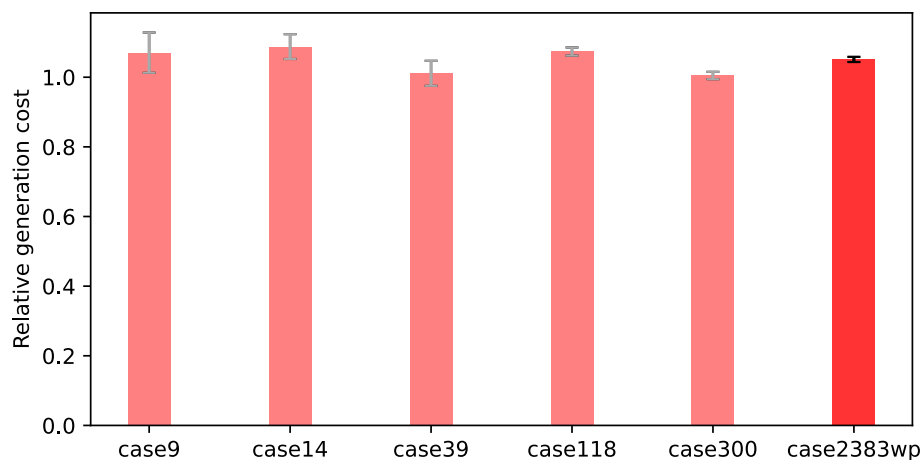


Figure 10 – Relative generation cost of a GNN model with regards to the IPS, when trained on *case2383wp* and tested on other power grids. The results obtained for *case2383wp* are highlighted.

With regards to computation time, Figure 11 demonstrates that the model outperforms the IPS by a considerable margin. This is especially noticeable for *case2383wp*, in which the GNN was 10^3 faster, given that its average computation time was approximately 6×10^{-2} s and that the IPS took around 90 s on average to calculate the OPF of the grid. Lastly, it should be once again emphasized that, regardless of the size of the grid, the computation time of the GNN model remained below 10^{-1} s, denoting much better scalability.

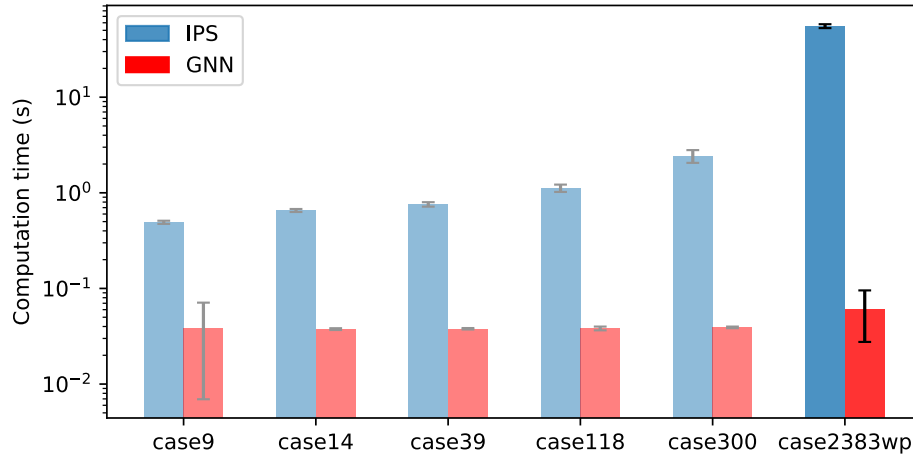


Figure 11 – Comparison of the computation time of the GNN model (red bars) and the IPS (blue bars), when trained on *case2383wp* and tested on other power grids. The results obtained for *case2383wp* are highlighted.

Being able to solve the OPF problem so fast could bring several benefits to the operation of power grids. For example, it could help system operators make timely decisions to minimize the overall operational costs and to ensure reliable grid operation, improving the economic efficiency of the power system. In addition to that, during emergency situations or contingencies, such as equipment failures, fast computation of the OPF would allow for the impact on the grid to be quickly assessed and dealt with, helping minimize downtime and secure the resilience of the system.

7 Final Remarks

This work explored the application of GNNs for solving the OPF problem in electrical grids. The experimental results demonstrated the potential of GNN-based models in addressing this complex optimization problem and highlighted the advantages they offer over traditional approaches such as the interior point method. Hence, further research on the use of GNNs to tackle other problems in the power systems domain could yield auspicious results.

7.1 Conclusion

In this paper, a generalizable and scalable GNN architecture was proposed to solve the OPF problem. As shown by the evaluation on diverse power grids, the model is able to compute the OPF of an electrical grid much faster than traditional methodologies. Thus, the main goals of this research (described in Section 1.2) were successfully achieved.

One notable feature of GNNs is their ability to capture and leverage the underlying graph structure of the power grid, enabling them to model the dependencies and interactions between different components. This graph-based representation enhances the generalization capability of the model, allowing it to adapt to and achieve good performance on various grid topologies, sizes and configurations.

Another significant benefit of using GNNs to solve the OPF problem refers to their scalability. By processing local neighborhood information iteratively, GNNs can handle computing the OPF for large power systems (with hundreds to thousands of buses) much faster than traditional numerical approaches. This scalability is essential for practical application in real-world power grids, where the number of buses, generators, and transmission lines can be substantial.

7.2 Contributions

Among the main contributions put forward as a result of this work, the following ones should be emphasized:

- **Computational speedup:** the experimental results showcase that the GNN model is able to compute solutions to the OPF problem much faster than traditional methods. Systems that operate under restrictive time limits could especially benefit from this speedup.

- **Generalization capability:** through its graph-based representation, the proposed architecture displays the capacity to to seamlessly adapt across diverse grid topologies and configurations (zero-shot property).
- **Scalability:** efficient processing of information from local neighbours enables the model to scale well across power systems with varying sizes, ranging from tens to even thousands of buses.

7.3 Future Perspectives

This work focused on proving the applicability of GNNs to solve the OPF problem. Future work could aim at improving the accuracy of the GNN by, e.g., performing a thorough hyperparameter tuning process, as the performance of GNN models can be sensitive to the choice of hyperparameters, such as learning rates, regularization parameters and batch size. Adjusting the training process, e.g., by modifying the dataset generation strategy, increasing the number of training samples, or experimenting with different loss functions, could also be an interesting way to achieve better results. Besides that, future work could focus on addressing the challenge of applying the methodology to real-world data, which are potentially more complex than the test cases used in this thesis, especially considering that the behaviour of actual power grids may evolve over time.

Bibliography

- BIENSTOCK, D.; VERMA, A. Strong NP-hardness of AC power flows feasibility. *Operations Research Letters*, v. 47, n. 6, p. 494–501, 2019. ISSN 0167-6377. Cited on page [13](#).
- BOYD, S.; VANDENBERGHE, L. *Convex Optimization*. USA: Cambridge University Press, 2004. Cited on page [33](#).
- CAIN, M.; O’NEILL, R.; CASTILLO, A. History of Optimal Power Flow and Formulations. *Federal Energy Regulatory Commission - Increasing Efficiency through Improved Software*, p. 1–31, dez. 2012. Cited on page [13](#).
- CASTILLO, A.; O’NEILL, R. Computational performance of solution techniques applied to the ACOF. *Federal Energy Regulatory Commission - Increasing Efficiency through Improved Software*, p. 1–34, fev. 2013. Cited on page [22](#).
- CHATZIVASILEIADIS, S. *Lecture Notes on Optimal Power Flow (OPF)*. 2018. Cited on page [13](#).
- COFFRIN, C. et al. PowerModels. JL: An Open-Source Framework for Exploring Power Flow Formulations. In: *2018 Power Systems Computation Conference (PSCC)*. [S.l.: s.n.], 2018. p. 1–8. Cited on page [20](#).
- DEFFERRARD, M.; BRESSON, X.; VANDERGHEYNST, P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2016. p. 3844–3852. Cited on page [17](#).
- DONNOT, B. et al. Fast Power system security analysis with Guided Dropout. In: *European Symposium on Artificial Neural Networks*. [S.l.: s.n.], 2018. Cited on page [14](#).
- DONON, B. et al. Neural networks for power flow: Graph Neural Solver. *Electric Power Systems Research*, v. 189, p. 106547, 2020. ISSN 0378-7796. Cited 8 times on pages [14](#), [15](#), [23](#), [27](#), [28](#), [31](#), [35](#), and [38](#).
- DONON, B. et al. Graph Neural Solver for Power Systems. In: *2019 International Joint Conference on Neural Networks (IJCNN)*. [S.l.: s.n.], 2019. p. 1–8. Cited 2 times on pages [27](#) and [30](#).
- FAN, C. et al. Static Security Assessment of Power System Considering Governor Nonlinearity. In: *2019 IEEE Innovative Smart Grid Technologies - Asia (ISGT Asia)*. [S.l.: s.n.], 2019. p. 128–133. Cited on page [13](#).
- FEY, M.; LENSSEN, J. E. Fast Graph Representation Learning with PyTorch Geometric. In: *ICLR 2019 Workshop on Representation Learning on Graphs and Manifolds*. [S.l.: s.n.], 2019. Cited on page [27](#).
- GILMER, J. et al. Neural Message Passing for Quantum Chemistry. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. [S.l.]: JMLR.org, 2017. p. 1263–1272. Cited on page [18](#).

- GRAINGER, J.; STEVENSON, W. *Power System Analysis*. [S.l.]: McGraw-Hill, 1994. Cited on page 13.
- HAMILTON, W. L.; YING, R.; LESKOVEC, J. Inductive Representation Learning on Large Graphs. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2017. p. 1025–1035. Cited on page 17.
- HANSEN, J. B.; ANFINSEN, S. N.; BIANCHI, F. M. Power Flow Balancing with Decentralized Graph Neural Networks. *IEEE Transactions on Power Systems*, Institute of Electrical and Electronics Engineers (IEEE), v. 38, n. 3, p. 2423–2433, maio 2023. Cited 2 times on pages 14 and 27.
- HARRIS, C. R. et al. Array programming with NumPy. *Nature*, Springer Science and Business Media LLC, v. 585, n. 7825, p. 357–362, set. 2020. Cited on page 27.
- HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*, v. 9, p. 1735–80, 12 1997. Cited on page 32.
- HUNTER, J. D. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, IEEE COMPUTER SOC, v. 9, n. 3, p. 90–95, 2007. Cited on page 27.
- LINCOLN, R. *PYPOWER*. 2011. Accessed: 2023-07-04. Disponível em: <<https://rwl.github.io/PYPOWER/api>>. Cited on page 37.
- LOPEZ-GARCIA, T. B.; DOMÍNGUEZ-NAVARRO, J. A. Graph Neural Network Power Flow Solver for Dynamical Electrical Networks. In: *2022 IEEE 21st Mediterranean Electrotechnical Conference (MELECON)*. [S.l.: s.n.], 2022. p. 825–830. Cited on page 14.
- LOW, S. H. Convex Relaxation of Optimal Power Flow — Part II: Exactness. *IEEE Transactions on Control of Network Systems*, v. 1, n. 2, p. 177–189, 2014. Cited on page 13.
- MILANO, F. *Power System Modelling and Scripting*. [S.l.]: Springer, 2010. Cited on page 20.
- OWERKO, D.; GAMA, F.; RIBEIRO, A. Optimal power flow using graph neural networks. In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2020. p. 5930–5934. Cited 2 times on pages 14 and 23.
- OWERKO, D.; GAMA, F.; RIBEIRO, A. *Unsupervised Optimal Power Flow Using Graph Neural Networks*. 2022. Cited 3 times on pages 14, 33, and 35.
- OZCANLI, A. K.; YAPRAKDAL, F.; BAYSAL, M. Deep learning methods and applications for electrical power systems: A comprehensive review. *International Journal of Energy Research*, v. 44, mar. 2020. Cited on page 13.
- PAN, X.; ZHAO, T.; CHEN, M. DeepOPF: Deep Neural Network for DC Optimal Power Flow. In: *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. [S.l.: s.n.], 2019. p. 1–6. Cited on page 14.

- PASZKE, A. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: *NIPS-W*. [S.l.]: Curran Associates, Inc., 2019. Cited on page 27.
- SALAM, M. A. *Fundamentals of Electrical Power Systems Analysis*. [S.l.]: Springer, 2020. Cited on page 13.
- SANTORO, A. et al. A simple neural network module for relational reasoning. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2017. p. 4974–4983. Cited on page 17.
- SCARSELLI, F. et al. The Graph Neural Network Model. *IEEE Transactions on Neural Networks*, v. 20, n. 1, p. 61–80, 2009. Cited on page 17.
- SINGH, M. K.; KEKATOS, V.; GIANNAKIS, G. B. Learning to Solve the AC-OPF Using Sensitivity-Informed Deep Neural Networks. *IEEE Transactions on Power Systems*, v. 37, n. 4, p. 2833–2846, 2022. Cited on page 14.
- ZIMMERMAN, R. D.; MURILLO-SÁNCHEZ, C. E.; THOMAS, R. J. MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education. *IEEE Transactions on Power Systems*, v. 26, n. 1, p. 12–19, 2011. Cited on page 36.
- ZOHRIZADEH, F. et al. A Survey on Conic Relaxations of Optimal Power Flow Problem. *European Journal of Operational Research*, v. 287, n. 2, p. 391–409, 2020. ISSN 0377-2217. Cited on page 13.